

# SIM8202G-M2 5G HAT (B)

## Features

- Integrates extensive network protocols, with multi drivers and software support, compatible with different OS including Windows / Linux / Android.
- USB 3.1 port (USB 2.0 compatible) for connecting to PC, Raspberry Pi, or Jetson Nano host board to enable high speed 5G communication.
- Standard M.2 B KEY slot, compatible with different 5G modules: RM500U-CN / RM500Q-GL / RM500Q-AE / RM502Q-AE series.
- Onboard UART, PWR, and RST control pin, built-in voltage level translator, enabled via DIP switch, for use with hosts like Raspberry Pi or Arduino.
- Onboard USB-C connector, enabled via a switch, for connecting standalone power supply for the module, allows more loads, stable and flexible power supply.
- Onboard power supply on/off switch, reset button and LED indicator, easy to turn on/off the module or monitor the operating status.
- 2 x SIM card slot, dual card single standby, switchable via AT command.
- High-efficiency power supply circuit, up to 3A output current.





## Specifications

	<b>SIM8202G-M2</b>
<b>Frequency band</b>	
Sub-6G (SA)	n1, n2, n3, n5, n7, n8, n12, n20, n28, n38, n40, n41, n48, n66, n71, n77, n78, n79
Sub-6G (NSA)	n41, n77, n78, n79

LTE-FDD	B1, B2, B3, B4, B5, B7, B8, B12, B13, B14, B17, B18, B19, B20, B25, B26, B28, B29, B30, B32, B66, B71
LTE-TDD	B34, B38, B39, B40, B41, B42, B43, B48
WCDMA	B1, B2, B3, B4, B5, B8
GNSS	GPS, GLONASS, Beidou, Galileo, and QZSS
<b>Data rate</b>	
Sub-6G	2.4 Gbps (DL) / 500 Mbps (UL)
LTE	1 Gbps (DL) / 200 Mbps (UL)
HSPA+	42 Mbps (DL) / 5.76 Mbps (UL)
<b>Software functions</b>	
Operating systems	Windows/Linux/Android
Communication protocols	TCP/IP, IPV4, IPV6, Multi-PDP, FTP, FTPS, HTTP, HTTPS, MQTTS, DNS, SSL3.0
SMS	Supports MT, MO, CB, Text, PDU
Firmware upgrade	Supports firmware upgrade via USB port

<b>Applications</b>	
Applicable regions	Regions with 5G Sub-6G signal coverage including China, US, Japan, South Korea, Europe, the Middle East, Latin America, etc.
Application examples	CPE, Smart gateway, Drone, Live streaming, Remote medical treatment, Intelligent security

## Selection Guide

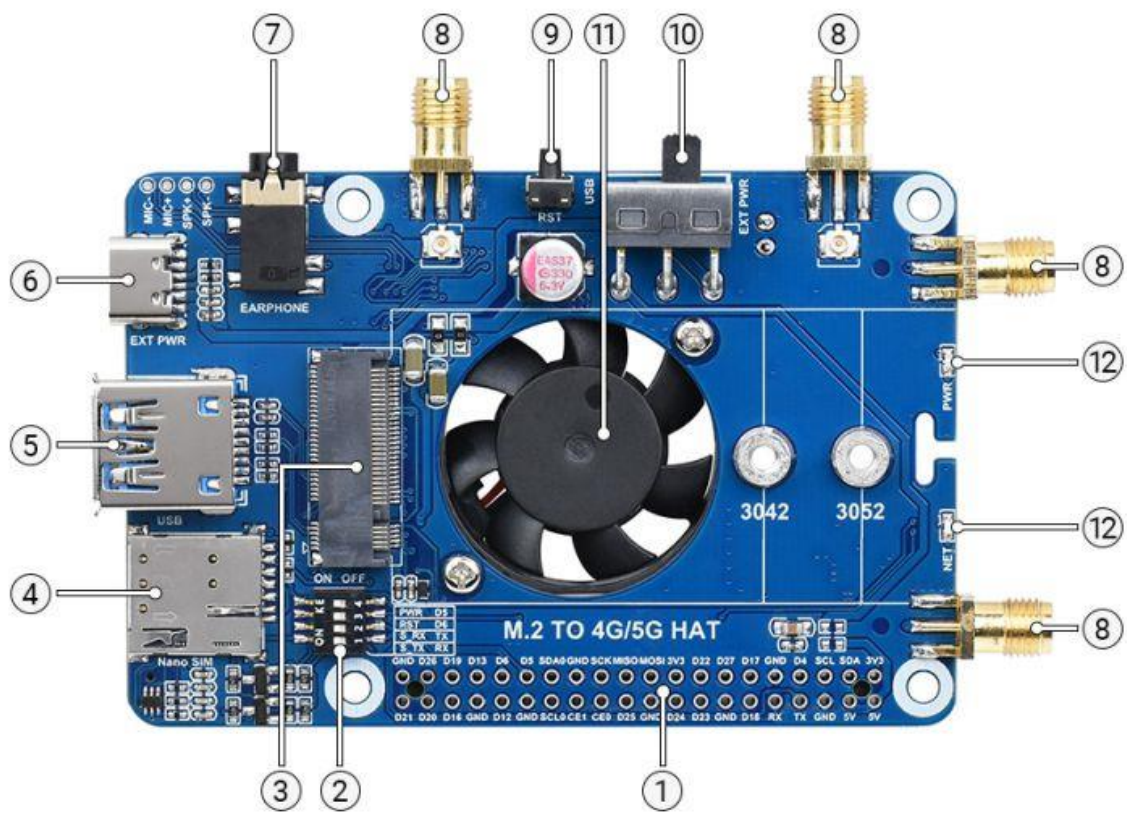
<b>5G Sub-6</b>	<b>RM500U-CN</b>	<b>RM500Q-GL</b>	<b>RM500Q-AE</b>	<b>RM502Q-AE</b>
				
<b>Region / Provider</b>	China	Global (except US)	Global (except China)	Global (except China)
<b>Operating Temperature</b>	-30 °C ~ +75 °C		-30 °C ~ +70 °C	
<b>Extension Temperature</b>	-40 °C ~ +85 °C			
<b>Dimensions</b>	30.0 × 52.0 × 2.3 (mm)			

<b>Weight</b>		8.9 (g)	8.7 (g)	
<b>Power Supply</b>		3.3~4.4 V, typical 3.7 V	3.135~4.4 V, typical 3.7 V	
<b>Power Consumption</b>		90 $\mu$ A @ shutdown  3.7 mA @ hibernate TBD @ USB 2.0, idle TBD @ USB 3.0, idle	70 $\mu$ A @ shutdown  4.0 mA @ hibernate 32 mA @ USB 2.0, idle 54 mA @ USB 3.0, idle	80 $\mu$ A @ shutdown  4.2 mA @ hibernate 39 mA @ USB 2.0, idle 54.5 mA @ USB 3.0, idle
<b>Frequency Band</b>				
<b>5G</b>	<b>5G NR NSA</b>	n41, n78, n79	n41, n77, n78, n79	n1, n2, n3, n5, n7, n8, n12, n20, n25, n28, n38, n40, n41, n48*, n66, n71, n77, n78, n79
	<b>5G NR SA</b>	n1, n28, n41, n77, n78, n79	n1, n2, n3, n5, n7, n8, n12, n20, n25, n28, n38, n40, n41, n48*, n66, n71, n77, n78, n79	
<b>LTE</b>	<b>LTE-FDD</b>	B1, B2, B3, B5, B7, B8, B20, B28	B1, B2, B3, B4, B5, B7, B8, B12(B17), B13, B14, B18, B19, B20, B25, B26, B28, B29, B30, B32, B66, B71	
	<b>LTE-TDD</b>	B34, B38, B39, B40, B41	B34, B38, B39, B40, B41, B42, B43, B48	
	<b>LAA</b>	-	B46	
<b>UMTS</b>	<b>WCDMA</b>	B1, B2, B5, B8	B1, B2, B3, B4, B5, B6, B8, B19	
<b>GNSS</b>		-	GPS / GLONASS / BeiDou (Compass) / Galileo	
<b>Data rate</b>				

<b>5G SA Sub-6</b>	downlink 2 Gbps; uplink 1 Gbps	downlink 2.1 Gbps; uplink 900 Mbps	downlink 2.1 Gbps; uplink 450 Mbps	downlink 4.2 Gbps; uplink 450 Mbps
<b>5G NSA Sub-6</b>	downlink 2.2 Gbps; uplink 575 Mbps	downlink 2.5 Gbps; uplink 600/650 Mbps	downlink 2.5 Gbps; uplink 650 Mbps	downlink 5 Gbps; uplink 650 Mbps
<b>LTE</b>	downlink 600 Mbps; uplink 150 Mbps	downlink 1.0 Gbps; uplink 200 Mbps	downlink 1.0 Gbps; uplink 200 Mbps	downlink 2 Gbps; uplink 200 Mbps
<b>UMTS</b>	downlink 42.2 Mbps; uplink 11 Mbps	downlink 42 Mbps; uplink 5.76 Mbps		

\* means developing/planning/processing

## What's On Board

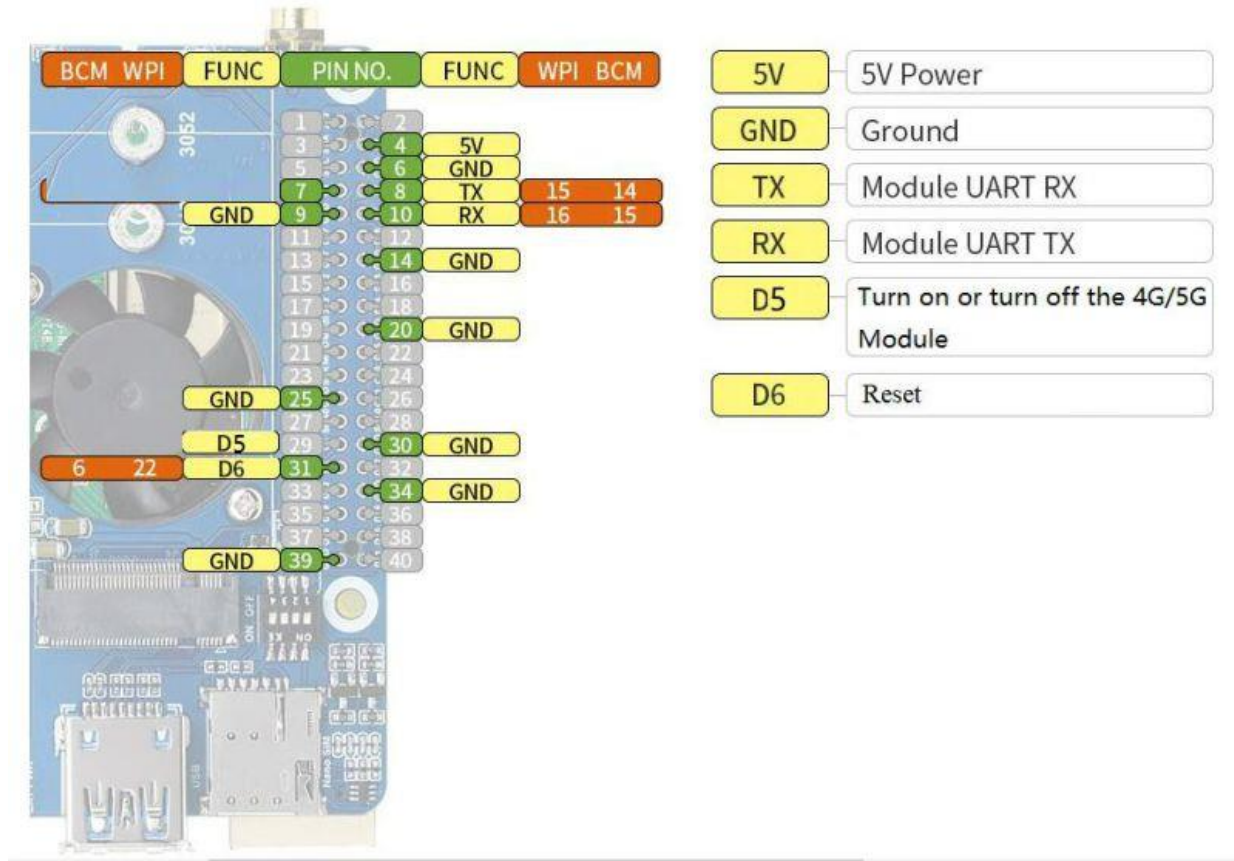


Number	Name	Description
①	Raspberry Pi GPIO interface	Easy connect to Raspberry Pi
②	Switch	Enable the corresponding pin
③	M.2 interface	Compatible with RM500U-CN / RM500Q-CN / RM500Q-GL /RM50XQ-AE and other series of 5G modules
④	SIM card holder	Onboard two SIM card slots, dual card single standby. The default SIM1 card slot works, SIM2 is on the back, requires module support, and must be switched through AT commands
⑤	USB3.1 interface	Backward compatible with USB 2.0, can be used to connect to PC/Raspberry Pi/Jetson Nano, etc.
⑥	USB Type-C interface	5V 3A input; stable and flexible power supply
⑦	Audio port	SIM82XX series support audio function, RM50XX series do not support this audio function
⑧	Antenna interface	Onboard four-way antenna, strong signal
⑨	Reset Switch	One-key reset
⑩	Power Switch	To facilitate the power supply mode of the control module: ——If set to USB, the module will provide power through the "⑤.USB3.1 interface"; ——If set to EXT PWR, the module will provide power through the "⑥.USB Type-C interface" external power supply
⑪	Cooling fan	Cool down the Raspberry Pi and 5G module at the same time
⑫	Indicator light	Check the module running status anytime, anywhere

- **Pinout Definition**

---

After connecting to Raspberry Pi, these pins (TX, RX, D4 and D6) can be connected or not through the DIP switch:



### 4G/5G modules function testing

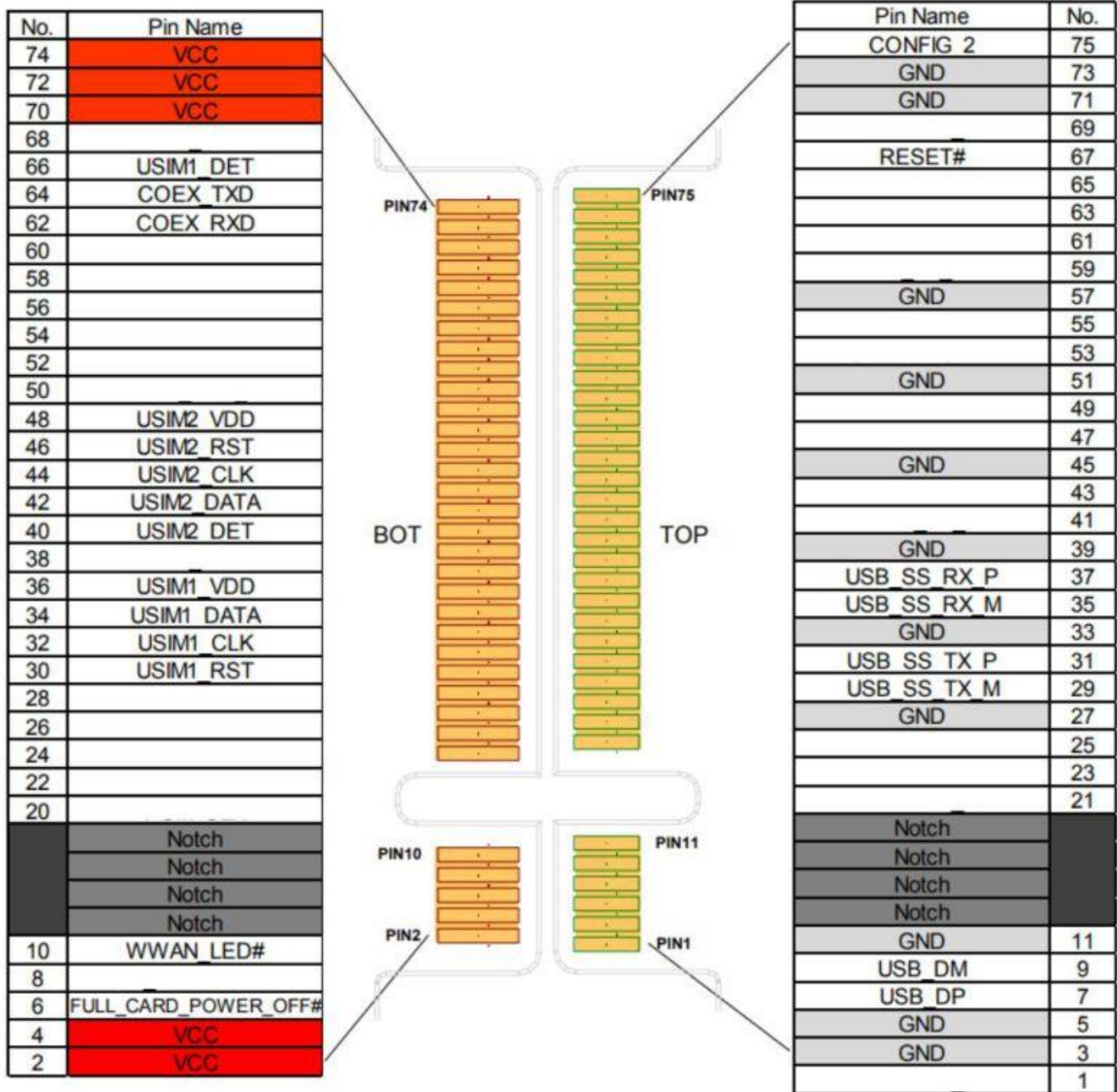
Category	4G/5G Module	Network Communication	GNSS Positioning	Voice calls through Earphone Port	Dual SIMs	UART Interface	External Power Supply?
5G	SIM8202G-M2	5G/4G/3G	Support	Support	Support	Support	Optional, but recommended

5G	SIM8200E A-M2	5G/4G/3G	Support	Support	Support	Support	Optional, but recommend ed
5G	RM500U- CN	5G/4G/3G	NOT Support	NOT Support	Support	Support	Recommend ed
5G	RM500Q- GL	5G/4G/3G	Support	NOT Support	Support	NOT Support	Recommend ed
5G	RM500Q- AE	5G/4G/3G	Support	NOT Support	NOT Support	NOT Support	Recommend ed
5G	RM502Q- AE	5G/4G/3G	Support	NOT Support	NOT Support	NOT Support	Recommend ed
LTE-A	EM06-E	LTE-A/4G/3G	NOT Support	NOT Support	NOT Support	NOT Support	Optional
LTE-A	A7906E	LTE-A/4G/3G	NOT Support	NOT Support	NOT Support	NOT Support	Optional
4G	SIM7600G -H-M2	4G/3G/2G	Support	Support	NOT Support	Support	Optional



## 4G/5G Module Compatibility

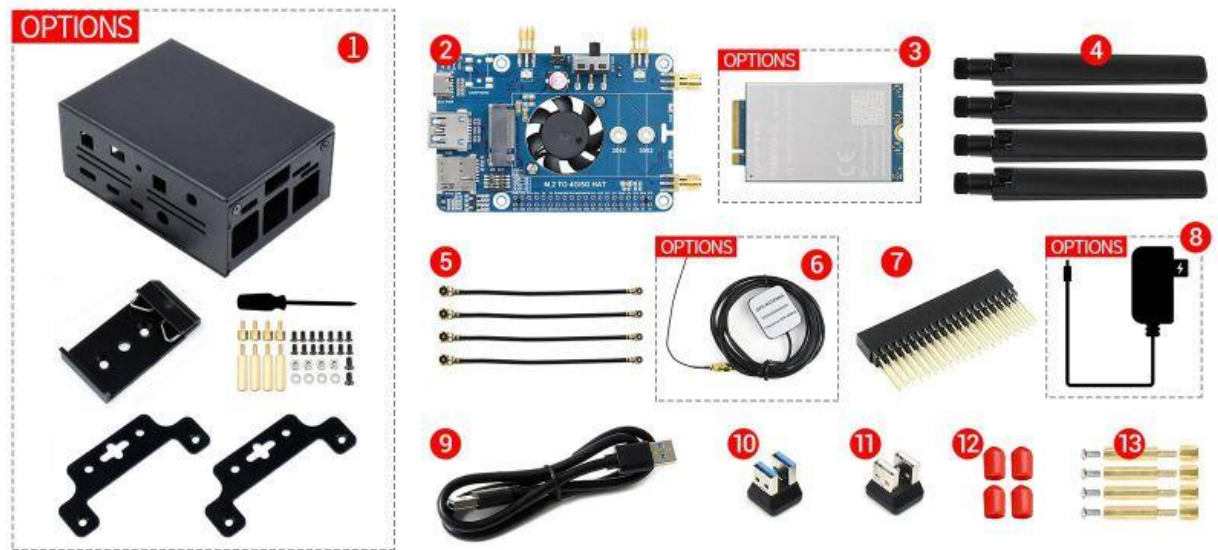
If you need to use the M.2 TO 4G/5G HAT for other 4G/5G modules, you can refer to the M.2 connection diagram below, check whether there is any pin conflict, and then connect to test:



## Working with Windows PC

### Hardware Preparation

- In addition to the items in the package,



- you need to prepare the following items:

- \* A 5G SIM card (no downtime and 5G enabled);
- \* A computer with a Windows operating system (Such as Windows 10)
- \* A headphone cable with a microphone (optional);

## Hardware Connection

Connect the 5G HAT with a usb3.0 cable, and connect an external 5V power supply to the Type-C power supply port of the 5G HAT, as shown in the figure:



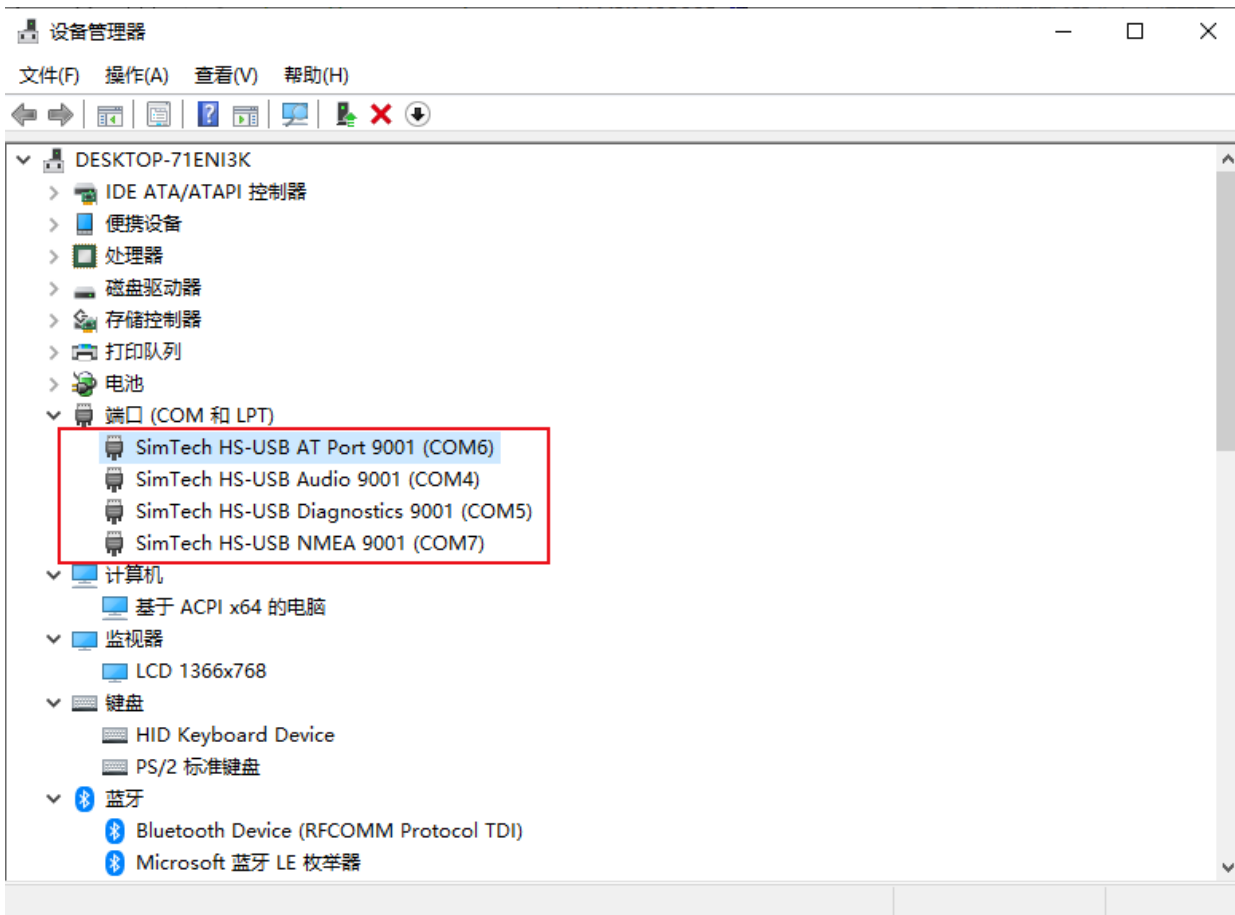
- **Install Driver**

---

Download the SIM8200 Driver from Resources part to your PC and unzip it.

Enter the SIM8200\_OS\_Driver\Windows directory

Enter the 1\_install directory and run the setup.exe file to install.



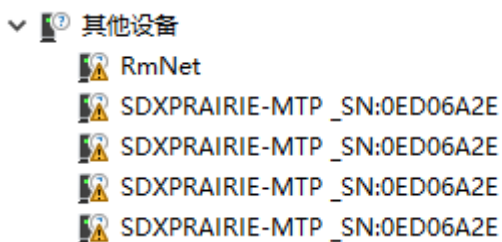
After connecting, a mobile network icon appears, you can disconnect other network and test the mobile network.

- **Install driver manually**

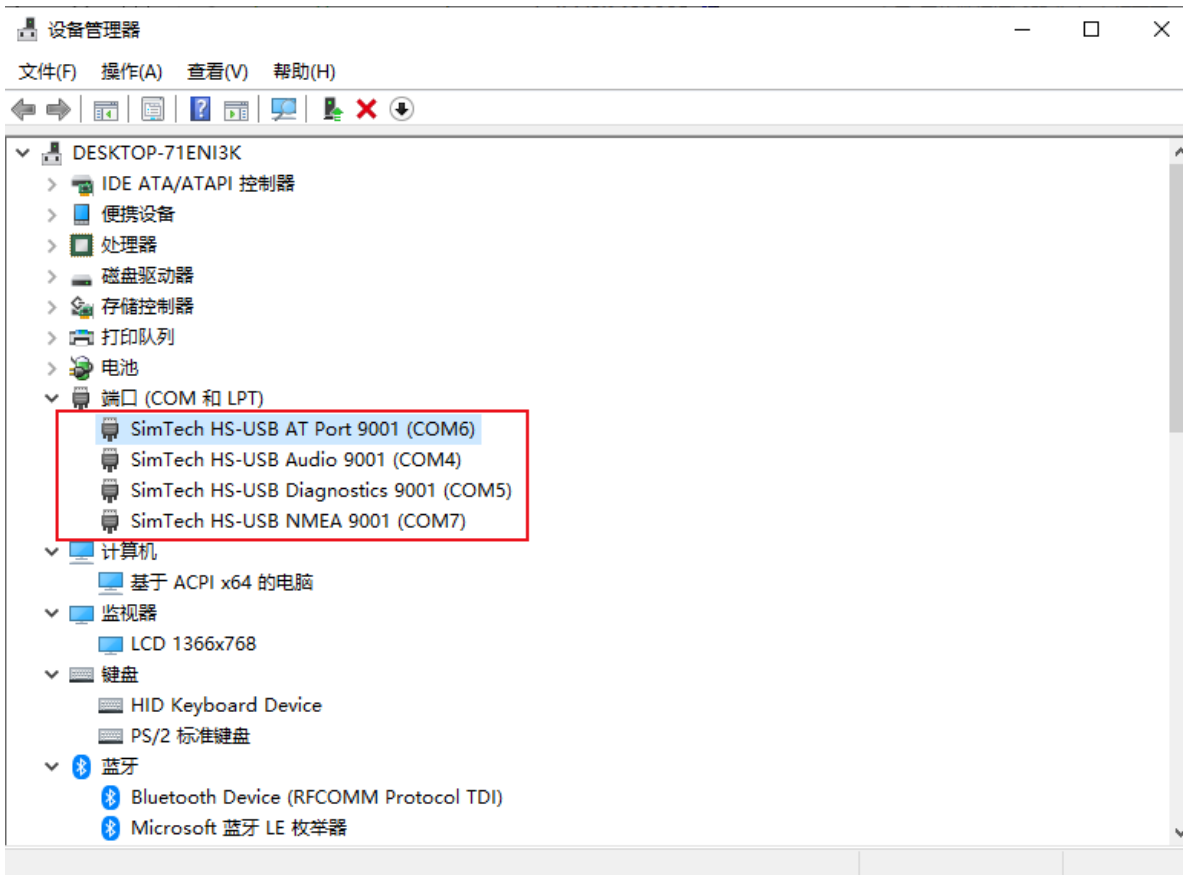
---

For some of PC, you need to add driver manually. If your PC doesn't recognized any COM ports in Device Manger, find unkown devices and upate the driver manually:

d Power on the 5G module and set the Switch into ON. four(it may be more than four) unkow devices are recognized by Windows PC.



Right-click the device , update the driver manually, choose SIM8200\_OS\_Driver\Windows, and then choose thd driver according to the version of your OS. You need to update for all the four/five/six devices.:

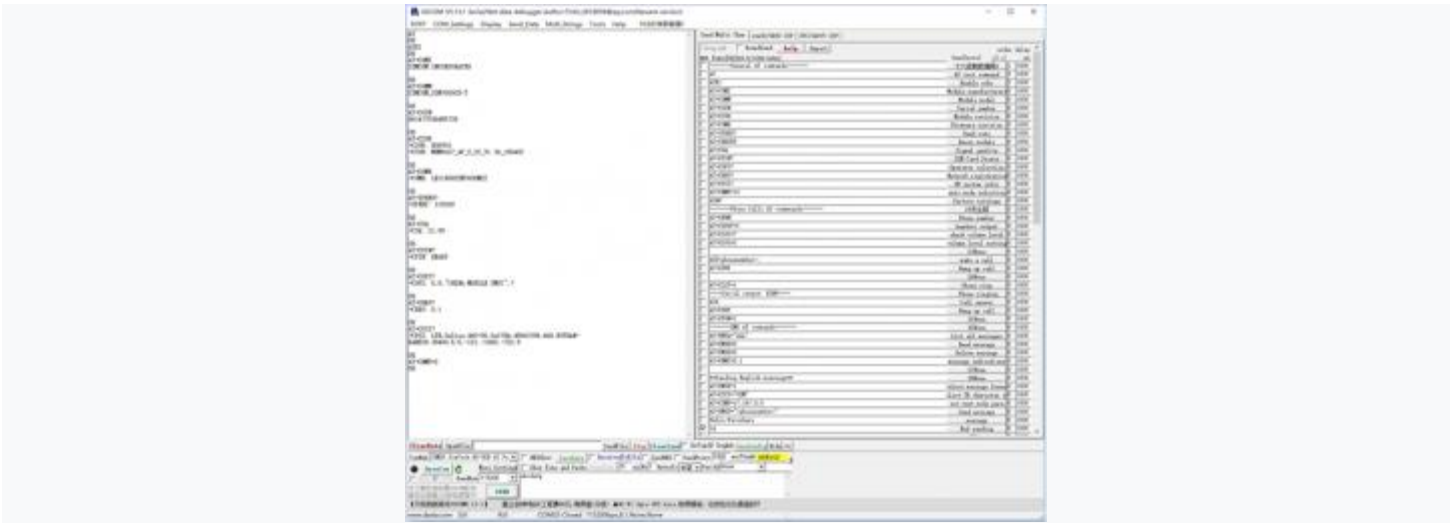


Four COM ports: AT is used for AT command controlling, Audio is used for dialing, Diagnostics is used for debugging, and NMEA is used for GPS.;



A mobile network will be setup automatically after updating, you can disconnect other network and test it.

## Common AT Command



- SIM7600X and SIM820X module supports AT command control, some basic AT commands are shown in the table below:

Command	Description	Return
AT	AT Test Command	OK
ATE	ATE1 set echo ATE0 close echo	OK
AT+CGMI	Query module manufacturer	OK
AT+CGMM	Query module model	OK
AT+CGSN	Query product serial number	OK
AT+CSUB	Query module version and chip	OK
AT+CGMR	Query the firmware version serial number	OK

AT+IPREX	Set the module hardware serial port baud rate	+IPREX: OK
AT+CRESET	reset module	OK
AT+CSQ	Network signal quality query, return signal value	+CSQ: 17,99 OK
AT+CPIN?	Query the status of the SIM card and return READY, indicating that the SIM card can be recognized normally	+CPIN: READY
AT+COPS?	Query the current operator, the operator information will be returned after normal networking	+COPS: OK
AT+CREG?	Query network registration status	+CREG: OK
AT+CPSI?	Query UE system information	
AT+CNMP?	Network mode selection command: 2: Automatic 13: GSM only 38: LTE only 48 : Any modes but LTE ... ..	OK

- Open the device manager, find the port number corresponding to AT Port; then open the sscm software, select the corresponding port and baud rate, check "Add carriage return and line feed"; click the sscm "Extension" button and pull out the

preset It is best to open the serial port and send the corresponding AT command to test.

## Choose SIM card

The 5G HAT has two SIM card slots onboard, dual SIM card and single standby, which can be switched and enabled by AT command

- SIM card 1 is selected by default, You can use the following command to query and confirm:

```
AT+SMSIMCFG?
```

- To switch SIM card 2, please use the following command:

```
AT+SMSIMCFG=1,2
```

- Switch back to the SIM card 1, please use the following command:

```
AT+SMSIMCFG=1,1
```

- Check whether the corresponding card slot recognizes the SIM card:

```
AT+CPIN?
```

- ## Manual NDIS dial-up Internet

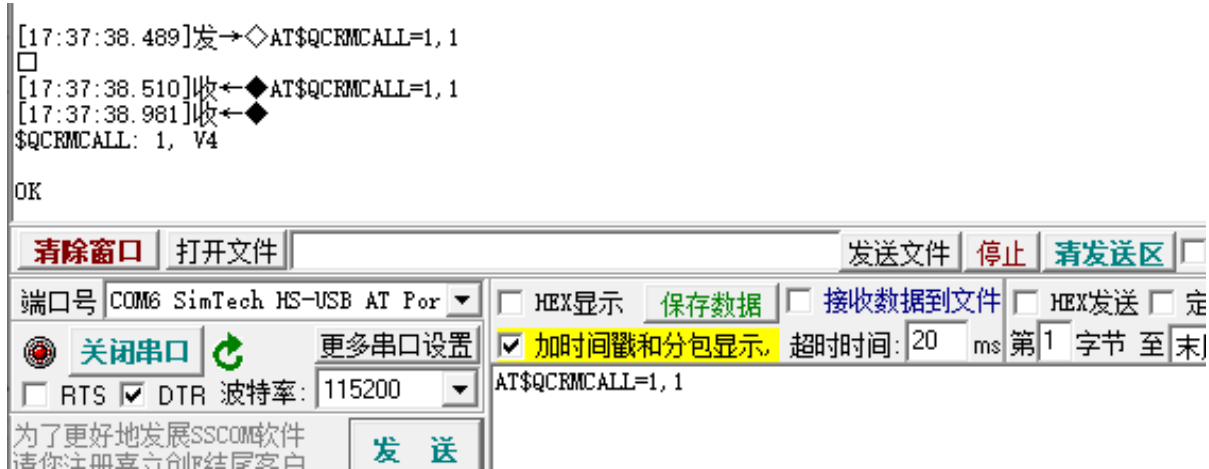
---

After installing the driver, some computers cannot automatically dial up to access the Internet, so you need to dial manually, the operation is as follows:

```
AT$QCRMCALL=1,1
```



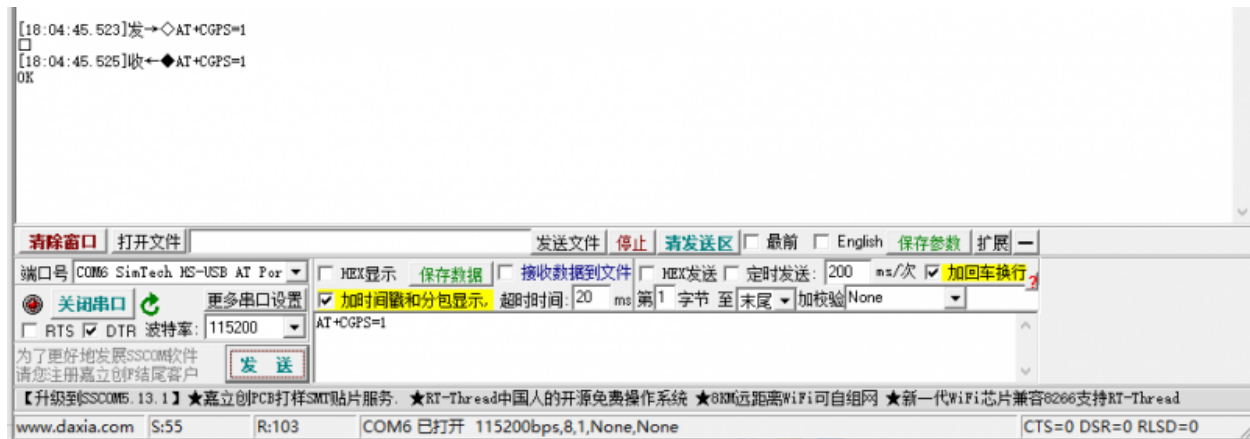
After dialing is successful, as shown in the figure below, the computer can go online normally



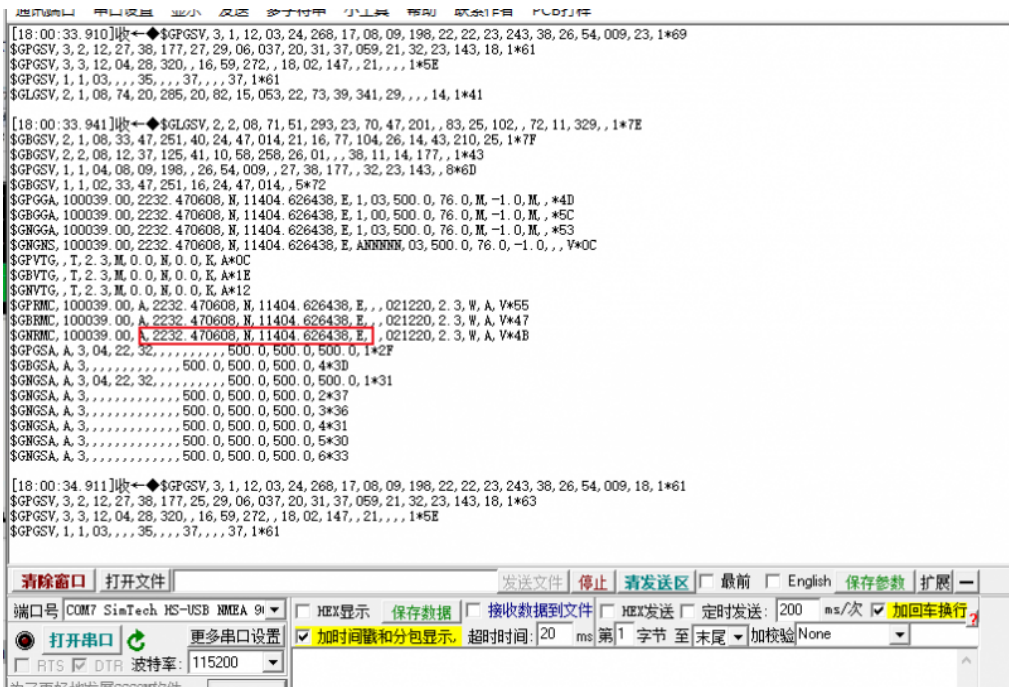
## • GPS Positioning

Connect the passive GPS antenna to the ANT5 of the module, and place the antenna outdoors facing the sky. Then send the AT command to turn on the GPS:

```
AT+CGPS=1
```



Now open the NEMA port, you can get GPS data:



Finally, turn off the GPS, you can use the AT command:




```
AT+CGPS=0
```

# Working with Raspberry Pi

## Hardware Connection

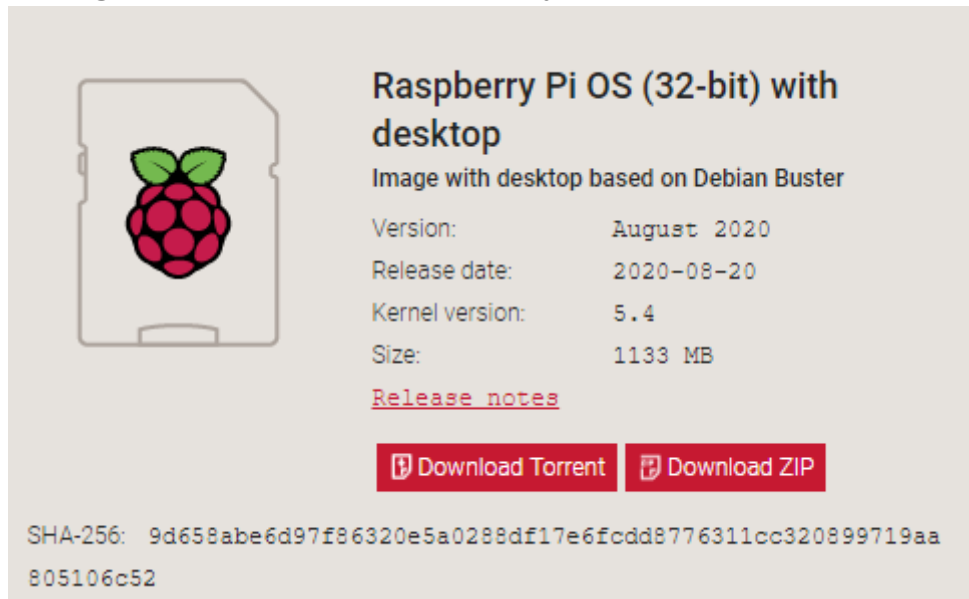
Connect the 5G HAT with a double-ended usb3.0 data cable, and connect an external 5V power supply to the Type-C power supply port of the 5G HAT, as shown in the figure:

Raspbe rry Pi	Pi 4B	Pi 3B/3B+	CM4-IO-BASE
USB adapter	USB3.0 adapter	USB2.0 adapter	USB3.0 adapter

<p>Connec tion</p>			
<p>Note</p>	<p>It is recommended to connect an external 5V power supply at the arrow.</p>		

## The use of Raspberry Pi OS

Please download the newest Raspberry Pi OS [2020-08-20-raspbian-buster-armhf](https://www.raspberrypi.org/downloads/raspbian/), All the settings are based on Kernel 5.4, if you use old version, please update first.



**Raspberry Pi OS (32-bit) with desktop**  
Image with desktop based on Debian Buster

Version: August 2020  
Release date: 2020-08-20  
Kernel version: 5.4  
Size: 1133 MB

[Release notes](#)

[Download Torrent](#) [Download ZIP](#)

SHA-256: 9d658abe6d97f86320e5a0288df17e6fcdd8776311cc320899719aa805106c52

- ### Configuration

This configuration is only needed at the first time.  
Open a terminal and run the following commands:

```
sudo apt-get install p7zip-full
wget https://www.waveshare.com/w/upload/f/fb/SIM8200-M2_5G_HAT_code.7z
```

```
7z x SIM8200-M2_5G_HAT_code.7z
sudo chmod 777 -R SIM8200-M2_5G_HAT_code
cd SIM8200-M2_5G_HAT_code
sudo ./install.sh
```

Please do not modify **option, qmi\_wwan\_simcom, default.script,install.sh files or directories**, otherwise the driver cannot be installed normally.

If you get error information, please check if you use the 2020-08-20-raspbian-buster-armhf OS and take screenshot of error information and contact our support team for help.

Run `ifconfig -a` command to check WWAN0 interface.

```
pi@raspberrypi:~$ ifconfig -a
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.3.131 netmask 255.255.255.0 broadcast 192.168.3.255
    inet6 fe80::d244:10b2:68b5:7245 prefixlen 64 scopeid 0x20<link>
    ether dc:a6:32:15:53:ae txqueuelen 1000 (Ethernet)
    RX packets 3432 bytes 749255 (731.6 KiB)
    RX errors 0 dropped 1 overruns 0 frame 0
    TX packets 2684 bytes 517939 (505.7 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 12 bytes 720 (720.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 12 bytes 720 (720.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

wlan0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.3.18 netmask 255.255.255.0 broadcast 192.168.3.255
    inet6 fe80::72f2:4bda:904:bedd prefixlen 64 scopeid 0x20<link>
    ether dc:a6:32:15:53:af txqueuelen 1000 (Ethernet)
    RX packets 392 bytes 60128 (58.7 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 469 bytes 93530 (91.3 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

wwan0: flags=4291<UP,BROADCAST,RUNNING,NOARP,MULTICAST> mtu 1500
    inet 169.254.160.22 netmask 255.255.0.0 broadcast 169.254.255.255
    inet6 fe80::b771:2b43:17d9:2ac6 prefixlen 64 scopeid 0x20<link>
    ether 22:3a:ef:1c:81:fa txqueuelen 1000 (Ethernet)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 14 bytes 3379 (3.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

- **Testing with AT command**

```
sudo apt-get install minicom
```

```
sudo minicom -D /dev/ttyUSB2
```

```
Welcome to minicom 2.7.1

OPTIONS: I18n
Compiled on Aug 13 2017, 15:25:34.
Port /dev/ttyUSB2, 15:02:32

Press CTRL-A Z for help on special keys

AT
OK
█
```

- ## 5G Networking

---

```
cd Goonline
make
sudo ./simcom-cm
```

After running codes, DNS information are shown as figure below:

```
root@raspberrypi:/home/pi/sim8200/Goonline# sudo ./simcom-cm
[06-16_04:32:16:413] SIMCOM_CM START...
[06-16_04:32:16:414] ./simcom-cm profile[1] = (null)/(null)/(null)/0, pincode = (null)
[06-16_04:32:16:421] Find qmichannel = /dev/cdc-wdm0
[06-16_04:32:16:422] Find usbnet_adapter = wwan0
[06-16_04:32:16:478] cdc_wdm_fd = 7
[06-16_04:32:16:801] Get clientWDS = 15
[06-16_04:32:16:833] Get clientDMS = 1
[06-16_04:32:16:865] Get clientNAS = 2
[06-16_04:32:16:897] Get clientUIM = 1
[06-16_04:32:16:929] Get clientWDA = 1
[06-16_04:32:16:961] requestBaseBandVersion MPSS.HI.2.0.5-00222.2-SDX55_CPEALL_PACK-1 1 [May 28 2020 17:00:00]
[06-16_04:32:17:025] requestGetSIMStatus SIMStatus: SIM_READY
[06-16_04:32:17:057] requestRegistrationState2 MCC: 460, MNC: 0, PS: Attached, DataCap: LTE
[06-16_04:32:17:089] requestQueryDataCall IPv4ConnectionStatus: DISCONNECTED
[06-16_04:32:17:153] requestRegistrationState2 MCC: 460, MNC: 0, PS: Attached, DataCap: LTE
[06-16_04:32:17:217] requestSetupDataCall WdsConnectionIPv4Handle: 0xf105d700
[06-16_04:32:17:281] requestQueryDataCall IPv4ConnectionStatus: CONNECTED
[06-16_04:32:17:314] ifconfig wwan0 up
[06-16_04:32:17:340] busybox udhcpc -f -n -q -t 5 -s /usr/share/udhcpc/default.script -i wwan0
udhcpc: started, v1.30.1
No resolv.conf for interface wwan0.udhcpc
udhcpc: sending discover
udhcpc: sending select for 10.142.169.227
udhcpc: lease of 10.142.169.227 obtained, lease time 7200
[06-16_04:32:18:007] /usr/share/udhcpc/default.script: Resetting default routes
SIOCDELRT: No such process
[06-16_04:32:18:048] /usr/share/udhcpc/default.script: Adding DNS 120.196.165.7
[06-16_04:32:18:048] /usr/share/udhcpc/default.script: Adding DNS 221.179.38.7
```

After connecting two SIM820X to Raspberry Pi through USB, two network cards—wwan0 and wwan1 can be recognized, and the two network cards can be dialed at the same time through the following commands: (The network speed cannot be superimposed)

```
sudo ./simcom-cm -i wwan0
sudo ./simcom-cm -i wwan1
```

```
wwan0: flags=4305<UP,POINTOPOINT,RUNNING,NOARP,MULTICAST> mtu 1500
    inet 10.1.73.159 netmask 255.255.255.192 destination 10.1.73.159
    unspec 00-00-00-00-00-00-00-00-00-00-00-00-00-00-00 txqueuelen 1000 (UNSPEC)
    RX packets 2 bytes 612 (612.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 47 bytes 7110 (6.9 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

wwan1: flags=4305<UP,POINTOPOINT,RUNNING,NOARP,MULTICAST> mtu 1500
    inet 10.1.224.47 netmask 255.255.255.224 destination 10.1.224.47
    unspec 00-00-00-00-00-00-00-00-00-00-00-00-00-00-00 txqueuelen 1000 (UNSPEC)
    RX packets 11 bytes 3100 (3.0 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 61 bytes 9682 (9.4 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

- **Note: If the IP cannot be obtained or the networking is not successful, use the following commands to obtain the IP and set the DNS networking:**

```
sudo dhclient -v wwan0
sudo route add -net 0.0.0.0 wwan0
```

- **Auto-run**

---

If you want to set the codes auto-run after booting, you can modify rc.local file:

```
sudo nano /etc/rc.local
```

Add the line to file as below:

```
sudo /home/pi/SIM8200-M2_5G_HAT_code/Goonline/simcom-cm &
```

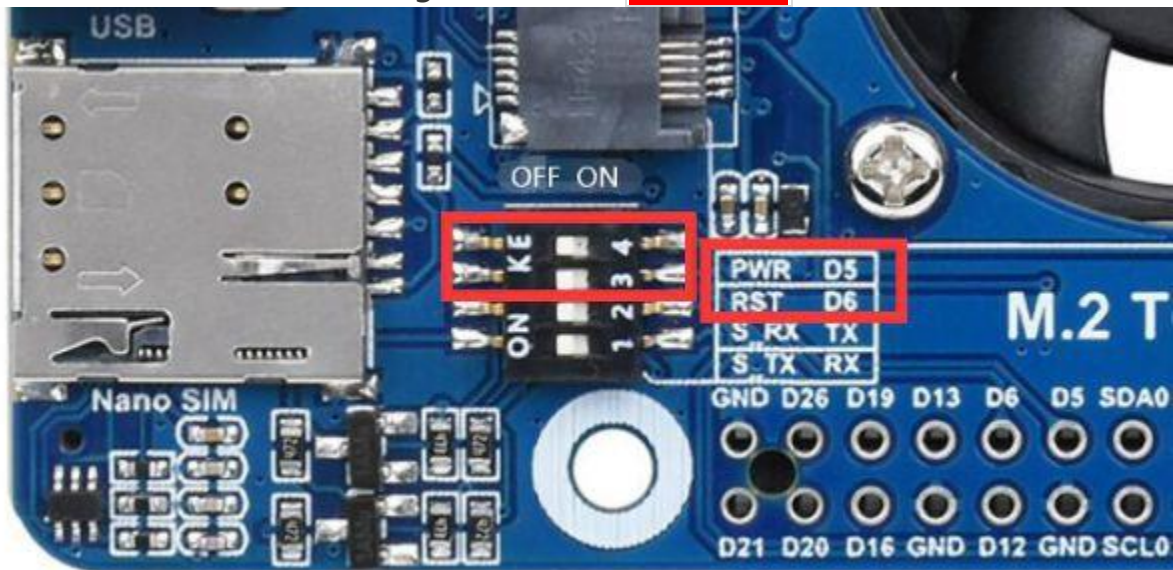
Note that you have to add "&" to the end of the command, make sure that the command can be run in the background, or the Pi may not boot normally.

- **5G HAT Power or Reset Control**

---

The 5G HAT is provided with PWR and RESET control pins, and the Raspberry Pi can control the module power on/off or reset through the high and low levels of these two pins.

- Turn the PWR and RST setting switches to **the right** to enable Pi control:



- **Power Control:**The following demonstrates the Shell script to control the PWR (D5 PIN) as an example:

```
cd /sys/class/gpio
echo 5 > export

cd gpio5
echo out > direction
echo 1 > value    #set to high level to shut down the 5G module
echo 0 > value    #set to low level to power on the 5G module
echo 5 > unexport
```

- **Reset Control:**The following demonstrates the Shell script to control the Reset (D6 PIN) as an example:

```
cd /sys/class/gpio
echo 6 > export

cd gpio6
echo out > direction
echo 1 > value    #set to high level to shut down the 5G module
sleep 1          #wait for 1 second
echo 0 > value    #set to low level to power on the 5G module
```

## The use of OpenWrt

- [Introduction to OpenWrt](#)

---

Soft routing is using desktops or servers and other equipment with software. It mainly depends on the settings of the software to achieve the functions of the router. The hard routing is a unique hardware device, including a processor, power supply, and embedded software to provide router functionality.

OpenWrt is a very popular soft routing system. It is a highly modular and highly automated embedded Linux system with powerful network components and scalability. It is often used in industrial control equipment, routers, and other equipment.

In addition to the functions of general home routers, OpenWrt soft routing can also achieve port forwarding, intranet penetration, 4G networking, FTP server and more powerful functions.

- [Burn the image](#)

---

Download the [RPI OpenWrt system](#), unzip the system in the Imgs directory, and use the burning tool to burn the system to the SD card.

- [Login and initial settings](#)

---

- After the OpenWrt system is turned on, and the Raspberry Pi is equivalent to a router. Therefore, use a network cable to connect the Raspberry Pi to the computer according to the use of the router (you can also use the mobile phone to search for WIFI, the default name is "OpenWrt").
- You can set the language to auto first.



OpenWrt AUTO REFRESH ON

Status  
System   
System   
Web Admin  
Administration  
Software  
TTYD Terminal  
Startup  
Scheduled Tasks  
Mount Points  
Disk Man  
LED Configuration  
Backup / Flash Firmware  
Custom Commands  
Scheduled Reboot  
FileTransfer  
Reboot  
Services  
NAS  
Network  
Bandwidth Monitor  
  
[Logout](#)

## System

Here you can configure the basic aspects of your device like its hostname or the timezone.

### System Properties

General Settings Logging **Language and Style**

Language auto   
Design 简体中文 (Simplified Chinese)

### Time Synchronization

Enable NTP client   
Provide NTP server

NTP server candidates

- ntp1.aliyun.com
- time1.cloud.tencent.com
- time.ustc.edu.cn
- cn.pool.ntp.org

SAVE & APPLY SAVE RESET

- Enter 192.168.1.1 on the web page, the default user name: root, and the default password: password, enter the OpenWrt web management interface

☆ 192.168.1.1

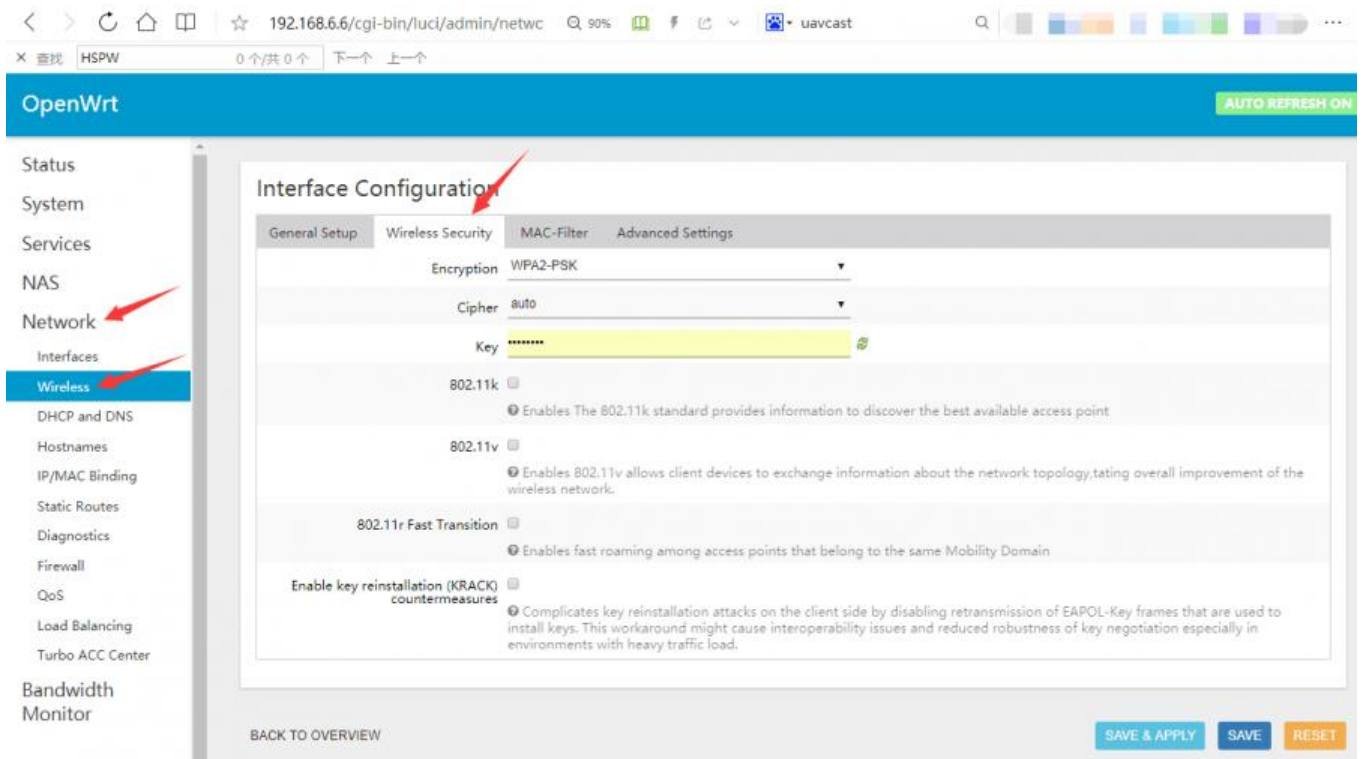
## Authorization Required

Please enter your username and password.

Username root  
Password password

LOGIN RESET

- Set WIFI password: Network —> Wireless —> interface configuration —> Wireless security



- Create the new interface: Network -> Interface -> Create interface
- Modify the IPv4 address of the lan port to a different IP that is not the same as the lan port IP of other routers in your home. (Many routers default the lan port IP to 192.168.1.1. If you do not modify the IP of the OpenWrt, it will easily lead to conflicts and fail to connect to the Internet)

If necessary, it is also recommended to disable the IPv6 allocation length. After the modification is completed, click "Save & Apply", and re-use 192.168.10.1 to access the OpenWrt console.

Status

System

Services

NAS

Network

Interfaces

Wireless

DHCP and DNS

Hostnames

IP/MAC Binding

Static Routes

Diagnostics

Firewall

QoS

Load Balancing

Turbo ACC Center

Bandwidth Monitor

Logout

## Interfaces - LAN

On this page you can configure the network interfaces. You can bridge several interfaces by ticking the "bridge interfaces" field and enter the names of several network interfaces separated by spaces. You can also use [VLAN](#) notation `INTERFACE.VLANID` (e.g.: `eth0.1`).

### Common Configuration

General Setup	Advanced Settings	Physical Settings	Firewall Settings
<b>Status</b> <span>Uptime: 2h 38m 28s</span>			
<b>MAC-Address:</b> DC:A6:32:E6:84:86			
<b>RX:</b> 1.39 MB (13676 Pkts.)			
<b>TX:</b> 3.33 MB (9951 Pkts.)			
<b>IPv4:</b> 192.168.6.6/24			
<b>IPv6:</b> fd3:8b03:5bb9::1/60			
br-lan			
Protocol Static address			
IPv4 address 192.168.10.1			
IPv4 netmask 255.255.255.0			
IPv4 gateway			
IPv4 broadcast			
Use custom DNS servers			
IPv6 assignment length 60			

- In addition, it is recommended to adjust the Firewall setting to connect the OpenWrt terminal and Web management interface through the local area.

Network —> Firewall, change all "reject" to "accept", click "Save & Apply" after modification, as shown in the picture below:

## Firewall - Zone Settings

The firewall creates zones over your network interfaces to control network traffic flow.

### General Settings

Enable SYN-flood protection

Drop invalid packets

Enable FullCone NAT

Input accept ▼

Output accept ▼

Forward reject ▼

### Zones

Zone → Forwardings	Input	Output	Forward	Masquerading	MSS clamping		
lan: <u>lan: [eth0]</u> ⇒ wan	<u>accept</u> ▼	<u>accept</u> ▼	<u>accept</u> ▼	<input type="checkbox"/>	<input type="checkbox"/>	EDIT	DELETE
wan:(empty) ⇒ ACCEPT	<u>accept</u> ▼	<u>accept</u> ▼	<u>accept</u> ▼	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	EDIT	DELETE

ADD

SAVE & APPLY SAVE RESET

- And then select System -> Administration, modify the allowed interface for SSH access to "unspecified" (that is, any interface can be accessed by ssh), check the Gateway port, and click "Save & Apply" after the modification is completed.

## SSH Access

Dropbear offers SSH network shell access and an integrated SCP server

### Dropbear Instance

DELETE

Interface lan: [eth0]

*\* unspecified*

Listen only on the given interface or, if unspecified, on all

Port 22

Specifies the listening port of this Dropbear instance

Password authentication

Allow SSH password authentication

Allow root logins with password

Allow the root user to login with password

Gateway ports

Allow remote hosts to connect to local SSH forwarded ports

ADD

At this point, you can connect to the OpenWrt web management interface or terminal through the IP address of the lan port or wan port.

- [Check the working status of the drive](#)

Connect to the OpenWrt terminal via SSH, and run the following commands to view the qmi driver, USB device, network port registration, and network port status:

```
dmesg | grep qmi
dmesg | grep ttyUSB
ls /dev | grep cdc-wdm
ifconfig wwan0
```

```
BusyBox v1.31.1 () built-in shell (ash)

-----
|_ _ | W I R E L E S S F R E E D O M
-----

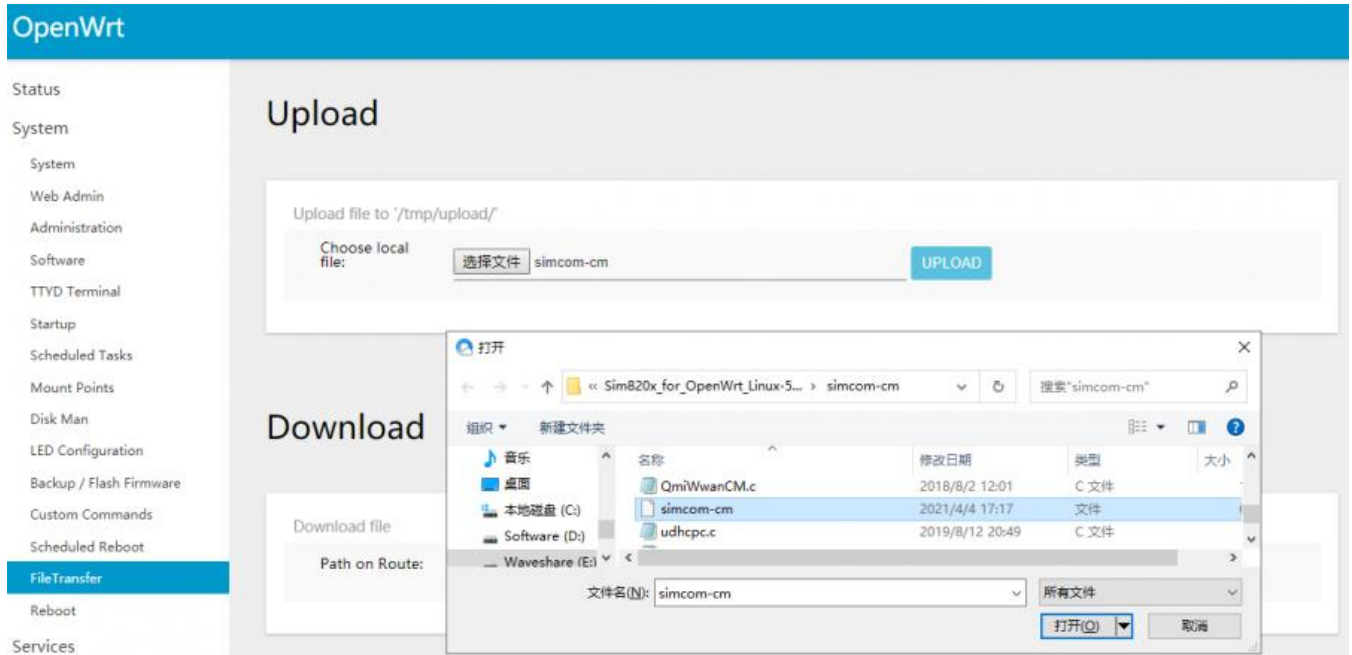
OpenWrt SNAPSHOT, r3015-faf254aed
-----

root@OpenWrt:~# dmesg | grep qmi
[ 7.457403] qmi_wwan_simcom 2-2:1.5: cdc-wdm0: USB WDM device
[ 7.464351] qmi_wwan_simcom 2-2:1.5: SIMCom 8200 work on RawIP mode
[ 7.473147] qmi_wwan_simcom 2-2:1.5 wwan0: register 'qmi_wwan_simcom' at usb-0000:01:00.0-2, WWA
[ 7.502085] usbcore: registered new interface driver qmi_wwan_simcom
root@OpenWrt:~# dmesg | grep ttyUSB
[ 1.607167] usb 2-2: GSM modem (1-port) converter now attached to ttyUSB0
[ 1.637509] usb 2-2: GSM modem (1-port) converter now attached to ttyUSB1
[ 1.743041] usb 2-2: GSM modem (1-port) converter now attached to ttyUSB2
[ 1.762854] usb 2-2: GSM modem (1-port) converter now attached to ttyUSB3
[ 2.064694] usb 2-2: GSM modem (1-port) converter now attached to ttyUSB4
root@OpenWrt:~# ls /dev | grep cdc-wdm
cdc-wdm0
root@OpenWrt:~# ifconfig wwan0
wwan0    Link encap:Ethernet  HWaddr F2:71:F4:34:13:E6
         BROADCAST NOARP MULTICAST  MTU:1500  Metric:1
         RX packets:0 errors:0 dropped:0 overruns:0 frame:0
         TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
         collisions:0 txqueuelen:1000
         RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

root@OpenWrt:~# █
```

## [Configure networking](#)

- Select System -> FileTransfer, select and upload simcom-cm in the simcom-cm directory of the folder:



Use the above "File Transfer" to upload, the uploaded program is located in "simcom-cm" under the /tmp/upload/ directory.

Enter the following commands in the terminal:

```
cp /tmp/upload/simcom-cm /  
chmod a+x simcom-cm  
./simcom-cm
```

**【Note】** : Closing this terminal will cause the networking program to stop, which will cause the network to be disconnected. It is recommended to run in the background.

```

root@OpenWrt:~# ./simcom-cm
[04-04_15:04:04:498] SIMCOM_CM START...
[04-04_15:04:04:498] ./simcom-cm profile[1] = (null)/(null)/(null)/0, pincode = (null)
[04-04_15:04:04:500] Find qmichannel = /dev/cdc-wdm0
[04-04_15:04:04:501] Find usbnet_adapter = wwan0
[04-04_15:04:04:510] cdc_wdm_fd = 7
[04-04_15:04:04:671] Get clientWDS = 15
[04-04_15:04:04:703] Get clientDMS = 1
[04-04_15:04:04:735] Get clientNAS = 2
[04-04_15:04:04:767] Get clientUIM = 1
[04-04_15:04:04:799] Get clientWDA = 1
[04-04_15:04:04:831] requestBaseBandVersion MPSS.HI.2.0.c3-00246-SDX55_CPEALL_PACK-1 1 [Oct 26 2020 16:00:00]
[04-04_15:04:04:895] requestGetSIMStatus SIMStatus: SIM_READY
[04-04_15:04:04:927] requestRegistrationState2 MCC: 460, MNC: 1, PS: Attached, DataCap: LTE
[04-04_15:04:04:959] requestQueryDataCall IPv4ConnectionStatus: DISCONNECTED
[04-04_15:04:05:023] requestRegistrationState2 MCC: 460, MNC: 1, PS: Attached, DataCap: LTE
[04-04_15:04:05:055] requestSetupDataCall WdsConnectionIPv4Handle: 0x8c2c71f0
[04-04_15:04:05:119] requestQueryDataCall IPv4ConnectionStatus: CONNECTED
[04-04_15:04:05:151] change mtu 1500 -> 1400
[04-04_15:04:05:151] ifconfig wwan0 up
[04-04_15:04:05:158] busybox udhcpc -f -n -q -t 5 -i wwan0
udhcpc: started, v1.31.1
udhcpc: sending discover
udhcpc: sending select for 10
udhcpc: lease of 10. obtained, lease time 7200
[04-04_15:04:05:250] udhcpc: ifconfig wwan0 10 netmask 255.255.255.252 broadcast +
[04-04_15:04:05:256] udhcpc: setting default routers: 10
[04-04_15:04:20:898] requestQueryDataCall IPv4ConnectionStatus: DISCONNECTED
[04-04_15:04:20:898] ifconfig wwan0 down
[04-04_15:04:20:962] requestRegistrationState2 MCC: 460, MNC: 1, PS: Attached, DataCap: LTE
[04-04_15:04:20:994] requestSetupDataCall QMUXResult = 0x1, QMUXError = 0xe
[04-04_15:04:26:050] requestRegistrationState2 MCC: 460, MNC: 1, PS: Attached, DataCap: LTE
[04-04_15:04:26:082] requestSetupDataCall WdsConnectionIPv4Handle: 0x8c2c71f0
[04-04_15:04:26:146] requestQueryDataCall IPv4ConnectionStatus: CONNECTED
[04-04_15:04:26:179] ifconfig wwan0 up

```

- At this time, open a new terminal of OpenWrt and enter the command: `ifconfig wwan0`. You can see that the `wwan0` network port has successfully obtained the operator IP and can ping the external network.

```

root@OpenWrt:~# ifconfig wwan0
wwan0    Link encap:Ethernet  HWaddr F2:
         inet addr:10.0.0.1          Bcast:10.0.0.255      Mask:255.255.255.0
         inet6 addr: fe80::f071:f4ff:fe34:13e6/64 Scope:Link
         UP BROADCAST RUNNING NOARP MULTICAST  MTU:1400  Metric:1
         RX packets:0  errors:0  dropped:0  overruns:0  frame:0
         TX packets:0  errors:0  dropped:0  overruns:0  carrier:0
         collisions:0 txqueuelen:1000
         RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

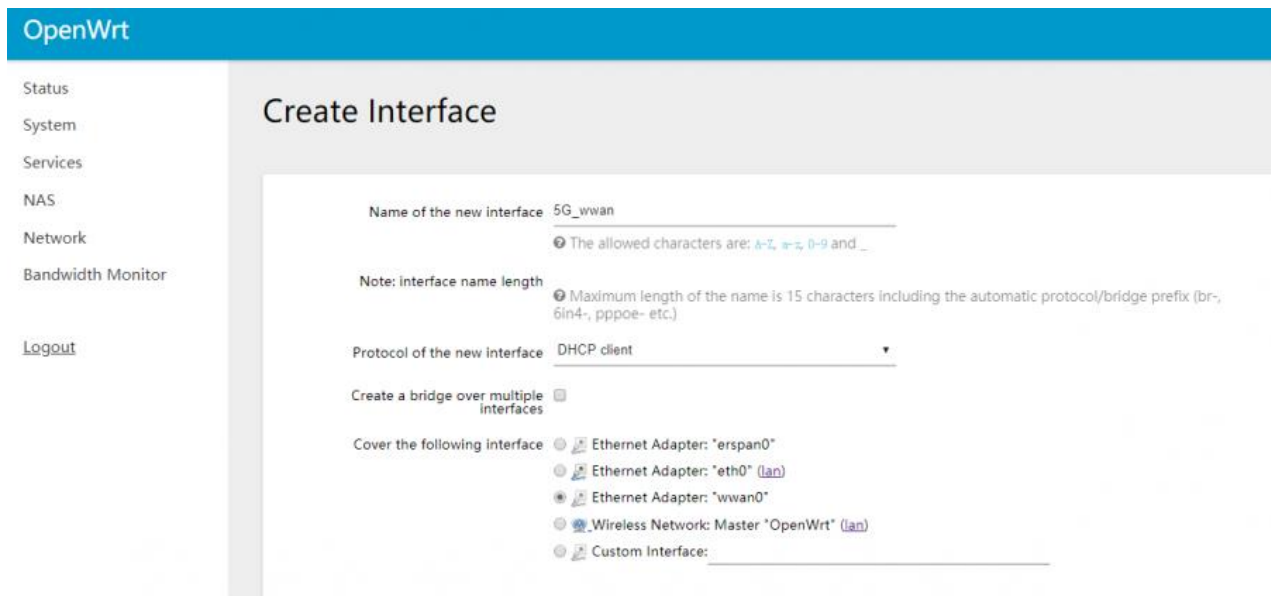
```

```

root@OpenWrt:~# ping 114.114.114.114
PING 114.114.114.114 (114.114.114.114): 56 data bytes
64 bytes from 114.114.114.114: seq=0 ttl=78 time=205.969 ms
64 bytes from 114.114.114.114: seq=1 ttl=69 time=62.752 ms
64 bytes from 114.114.114.114: seq=2 ttl=83 time=46.468 ms
64 bytes from 114.114.114.114: seq=3 ttl=71 time=35.838 ms
64 bytes from 114.114.114.114: seq=4 ttl=72 time=166.026 ms
64 bytes from 114.114.114.114: seq=5 ttl=75 time=115.396 ms
64 bytes from 114.114.114.114: seq=6 ttl=80 time=80.107 ms
^C
--- 114.114.114.114 ping statistics ---
7 packets transmitted, 7 packets received, 0% packet loss
round-trip min/avg/max = 35.838/101.793/205.969 ms
root@OpenWrt:~#

```

- Enter the Web management interface of OpenWrt, click Network —> Interface —> Create a new interface





- Enter the interface as shown in the picture below and confirm that the interface selection in "Physical Settings" is "wwan0".

The screenshot shows the OpenWrt web interface for configuring network interfaces. The page title is "Interfaces - 5G\_WWAN". The left sidebar shows the "Network" menu with "Interfaces" selected. The main content area has tabs for "General Setup", "Advanced Settings", "Physical Settings", and "Firewall Settings". The "Physical Settings" tab is active, showing a "Bridge interfaces" section with a checkbox and a list of interfaces. The "Interface" list includes "Ethernet Adapter: 'erspan0'", "Ethernet Adapter: 'eth0' (lan)", "Ethernet Adapter: 'wwan0' (5G\_wwan)", "Wireless Network: Master 'OpenWrt' (lan)", and "Custom Interface:". A red arrow points to the "wwan0" option.

- Confirm that the interface selection in "Firewall Settings" is "wan"

The screenshot shows the OpenWrt web interface for configuring network interfaces. The page title is "Interfaces - 5G\_WWAN". The left sidebar shows the "Network" menu with "Interfaces" selected. The main content area has tabs for "General Setup", "Advanced Settings", "Physical Settings", and "Firewall Settings". The "Firewall Settings" tab is active, showing a "Create / Assign firewall-zone" section. The "lan" zone is selected and highlighted in green, and the "wan" zone is highlighted in red with a red arrow pointing to it. The "unspecified -or- create:" field is also visible.

- Click "Save & Apply" to complete the network port settings, then return to the interface below, network-interface, you can see that the network port has been correctly identified.

Then the other devices can be connected to the OpenWrt wireless "OpenWrt" or through the network cable to connect to OpenWrt's own network port for networking.

## 5G network speed test

In the speed measurement part, because the Raspberry Pi comes with a Gigabit Ethernet port, and there are few USB network cards above Gigabit, we use the SpeedTest For Python tool to perform speed measurement by the commands. Connect to the terminal of OpenWrt and enter the commands one by one to measure the speed:

```
## Raspberry Pi OS
sudo apt install speedtest-cli
speedtest      # or use speedtest_cli
```

or

```
## OpenWRT
opkg update
opkg install python3
opkg install python3-pip
pip install speedtest_cli
speedtest      # or use speedtest_cli
```

## 5G HAT Minicom Serial Port Debugging

## 4G/5G Modules UART Interface Support

SIM8202G-M2, RM500U-CN, and SIM7600G-H-M2 Support UART Interface, Other 4G/5G Modules NOT Support!

For SIM8202G-M2 and SIM8200EA-M2 these two 5G modules, the UART interface is closed by default and needs to be opened through AT commands:

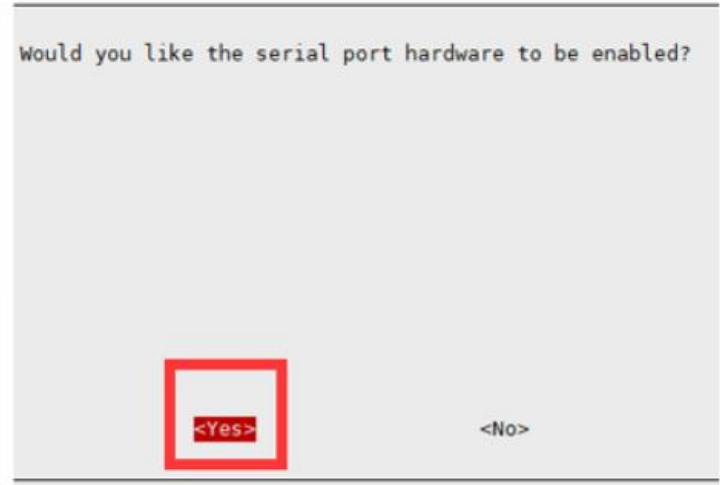
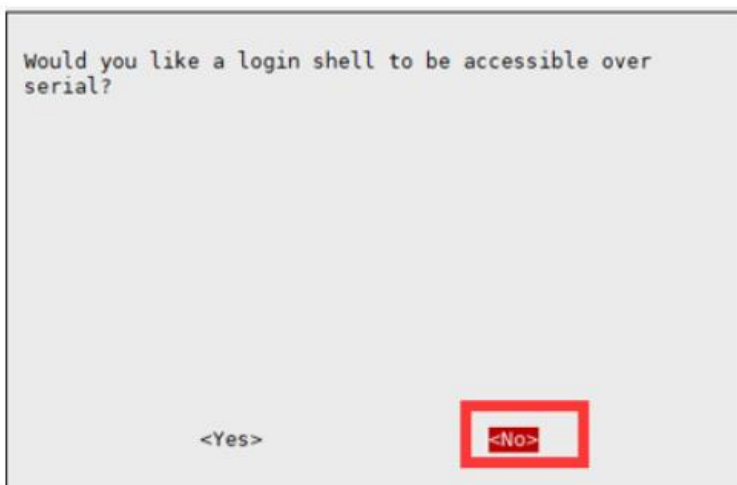
```
AT+CCUART=1
```

## Raspberry Pi Serial Port Enable

Since the Raspberry Pi serial port is used for terminal debugging by default, if you need to use the serial port, you need to modify the Raspberry Pi settings. Execute the following command to enter the Raspberry Pi configuration:

```
sudo raspi-config
```

Select Interfacing Options ->Serial ->NO ->YES to disable serial debugging

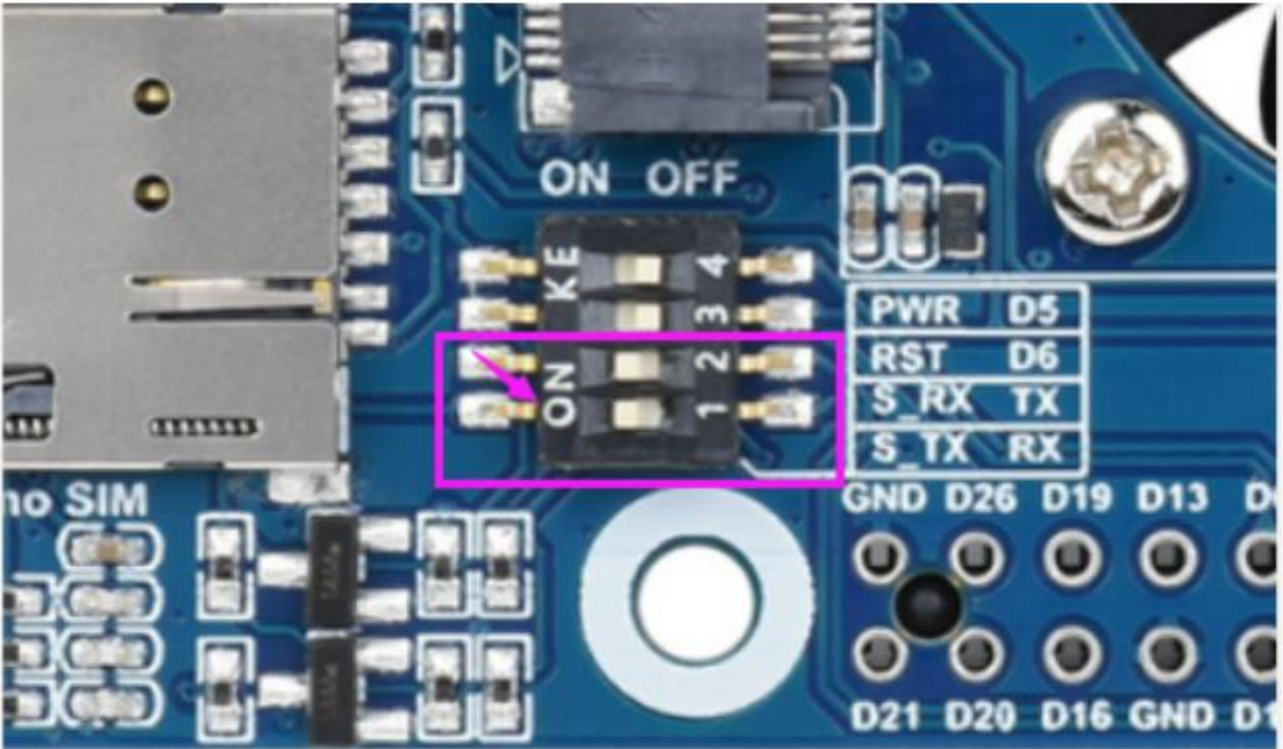


Open the /boot/config.txt file, find the following configuration statement to enable the serial port, if not, add it at the end of the file:

```
enable_uart=1
```

Restart to take effect

1. Insert the module into the Raspberry Pi, and turn the S\_TX and S\_RX of the DIP switch to ON:



2. Install minicom, minicom is a serial debugging tool for linux platform:

```
sudo apt-get install minicom
```

3. Open ttyS0 through minicom——ttyS0 is the serial port of Raspberry Pi 3B/3B+/4B, the default baud rate is 115200;

```
sudo minicom -D /dev/ttyS0
```

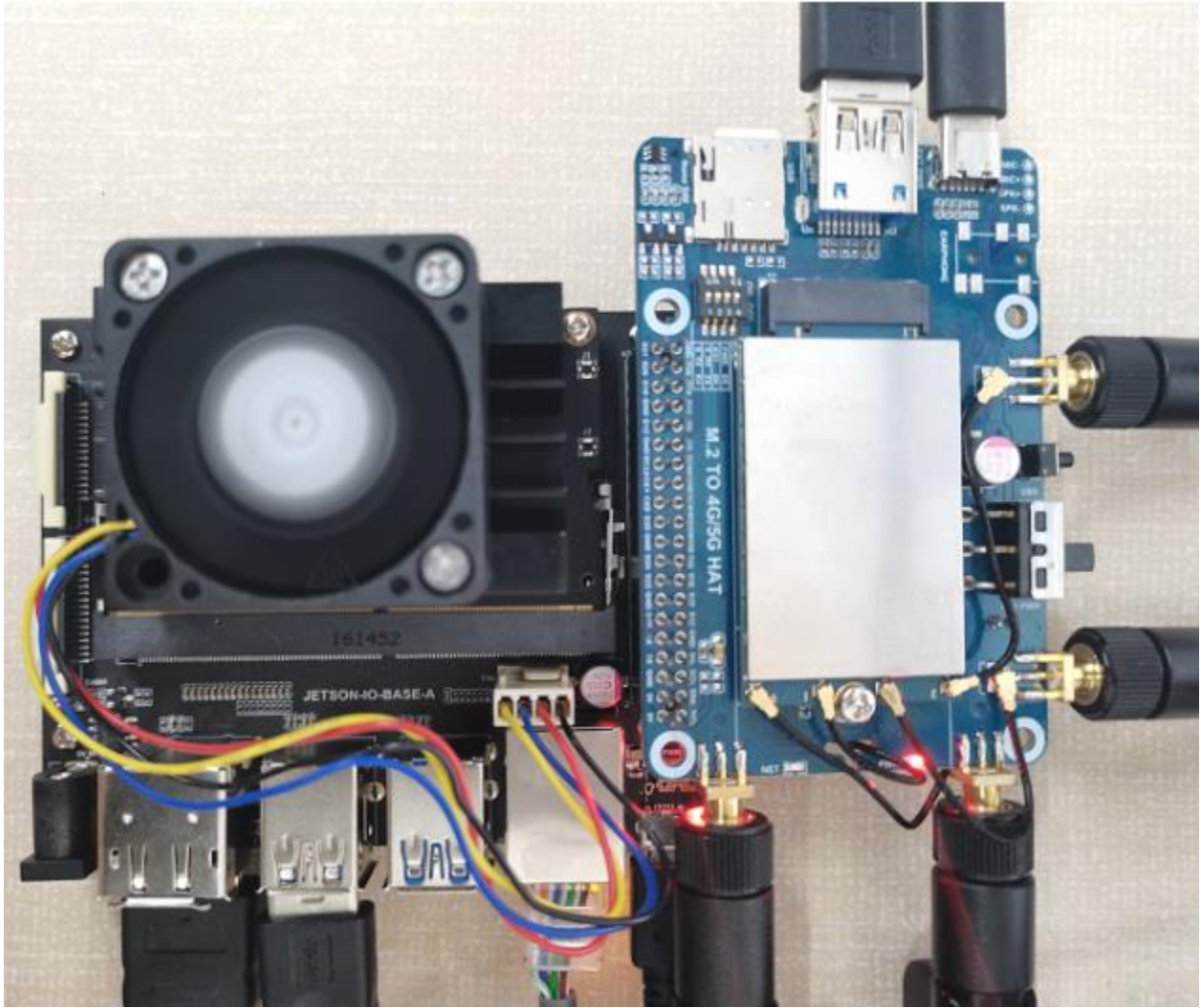
4. For Raspberry Pi 2B/zero, the user serial device number is ttyAMA0; you can use the following command line to confirm, serial0 is the selected serial device number, as shown below:

```
pi@raspberrypi:~$ ls -l /dev/serial*
lrwxrwxrwx 1 root root 5 Jul  7 08:35 /dev/serial0 -> ttyS0
lrwxrwxrwx 1 root root 7 Jul  7 08:35 /dev/serial1 -> ttyAMA0
```

## Working with Jetson Nano

### Hardware Connection

Connect the 5G HAT with a double-ended usb3.0 data cable, and connect an external 5V power supply to the Type-C power supply port of the 5G HAT, as shown in the figure:



We recommend you use the new [jetson-nano-sd-card-image](#), whose kernel version is higher than 4.9.140-tegra.

If you run with other Linux systems, please download the driver which is located in the SIM8200\_OS\_Driver\linux folder, and porting it according to the attached document.

## Setup software

---

Open a terminal and run the following command.

```
sudo apt-get install p7zip-full
wget https://www.waveshare.com/w/upload/0/07/Sim8200_for_jetsonnano.7z
7z x Sim8200_for_jetsonnano.7z -r -o./Sim8200_for_jetsonnano
sudo chmod 777 -R Sim8200_for_jetsonnano
cd Sim8200_for_jetsonnano
```

```
sudo ./install.sh
```

Please do not modify or delete the option, qmi\_wwan\_simcom, default.script, and install.sh folders, otherwise, the driver cannot be installed normally.

If the driver failed to install, please use 4.9.140-tegra version and try it again.

Run the command `ifconfig -a` to check WWAN0 port

```
wangkg@wangkg-desktop:~/test/Sim8200_for_jetsonnano$ sudo ./install.sh
Linux wangkg-desktop 4.9.140-tegra #1 SMP PREEMPT Thu Jun 25 21:25:44 PDT 2020 aarch64 aarch64 aarch64 GNU/Linux
[ 19.000331] usb 2-1.1: GSM modem (1-port) converter now attached to ttyUSB0
[ 19.000712] usb 2-1.1: GSM modem (1-port) converter now attached to ttyUSB1
[ 19.016545] usb 2-1.1: GSM modem (1-port) converter now attached to ttyUSB2
[ 19.016853] usb 2-1.1: GSM modem (1-port) converter now attached to ttyUSB3
[ 19.017280] usb 2-1.1: GSM modem (1-port) converter now attached to ttyUSB4
[ 144.041205] usbcore: registered new interface driver qmi_wwan_simcom
```

## Test AT command

---

```
sudo apt-get install minicom
```

```
sudo minicom -D /dev/ttyUSB2
```

```
Welcome to minicom 2.7.1

OPTIONS: I18n
Compiled on Aug 13 2017, 15:25:34.
Port /dev/ttyUSB2, 15:02:32

Press CTRL-A Z for help on special keys

AT
OK
```

## 5G Networking

---

```
cd Goonline
```

```
make
```

```
sudo ./simcom-cm
```

## Check if it generates the DNS

```
wangkg@wangkg-desktop:~/wangkg/sim8200/Goonline$ sudo ./simcom-cm
[10-30_11:47:11:974] Build Version: 2020-07-14 10:20:15
[10-30_11:47:11:975] SIMCOM_CM START...
[10-30_11:47:11:975] ./simcom-cm profile[1] = (null)/(null)/(null)/0, pincode = (null)
[10-30_11:47:11:977] Find /sys/bus/usb/devices/2-1.1 idVendor=1e0e idProduct=9001
[10-30_11:47:11:977] Find /sys/bus/usb/devices/2-1.1:1.5/net/wwan0
[10-30_11:47:11:977] Find usbnet_adapter = wwan0
[10-30_11:47:11:977] Find /sys/bus/usb/devices/2-1.1:1.5/usbmisc/cdc-wdm0
[10-30_11:47:11:977] Find qmichannel = /dev/cdc-wdm0
[10-30_11:47:11:978] netcard driver = qmi_wwan_simcom
[10-30_11:47:12:012] cdc_wdm_fd = 7
[10-30_11:47:13:013] QmiThreadSendQMITimeout pthread_cond_timeout_np=110, errno: 22 (Invalid argument)
[10-30_11:47:14:312] Get clientWDS = 15
[10-30_11:47:14:344] Get clientDMS = 1
[10-30_11:47:14:376] Get clientNAS = 2
[10-30_11:47:14:408] Get clientUIIM = 1
[10-30_11:47:14:440] Get clientWDA = 1
[10-30_11:47:14:472] requestBaseBandVersion MPSS.HI.2.0-00826.4-SDX55_CPEALL_PACK-1 1 [Jul 01 2020 00:00:00]
[10-30_11:47:14:536] requestGetSIMStatus SIMStatus: SIM_READY
[10-30_11:47:14:568] requestGetProfile[1] //0
[10-30_11:47:14:600] requestRegistrationState2 MCC: 460, MNC: 0, PS: Attached, DataCap: LTE
[10-30_11:47:14:632] requestQueryDataCall IPv4ConnectionStatus: DISCONNECTED
[10-30_11:47:14:696] requestRegistrationState2 MCC: 460, MNC: 0, PS: Attached, DataCap: LTE
[10-30_11:47:14:728] requestSetupDataCall WdsConnectionIPv4Handle: 0x531f2d60
[10-30_11:47:14:792] requestQueryDataCall IPv4ConnectionStatus: CONNECTED
[10-30_11:47:14:824] ifconfig wwan0 up
[10-30_11:47:14:849] busybox udhcpc -f -n -q -t 5 -s /usr/share/udhcpc/default.script -i wwan0
udhcpc: started, vl.27.2
udhcpc: sending discover
udhcpc: sending select for 10.43.52.87
udhcpc: lease of 10.43.52.87 obtained, lease time 7200
[10-30_11:47:15:113] /usr/share/udhcpc/default.script: Resetting default routes
SIOCDELRT: No such process
[10-30_11:47:15:126] /usr/share/udhcpc/default.script: Adding DNS 120.196.165.7
[10-30_11:47:15:126] /usr/share/udhcpc/default.script: Adding DNS 221.179.38.7
```

## Resources

### Assembly drawing

- [Case Assembly drawing](#)

### Demo Codes

- [SIM8200-M2 5G HAT Demo codes](#)
- [RPI OpenWrt System](#)

### Software

- [SIM8200 Driver](#)
- [SSCOM Software](#)
- [GPS Software](#)
- [Xshell software](#)
- [Unicode software](#)

## Document

- [Slm8200 Series AT Command Manual](#)
- [SIM8200EA-M2 Hardware Design Manual](#)
- [SIMcom SIM8200EA-M2 resources](#)
- [SIMcom SIM8202G-M2 resources](#)
- [SIM820X RNDIS Dial-Up](#)

### Related application cases

- [SIM820X 5G HAT is equipped with Raspbian Pi to open hotspots](#)
- [SIM820X PPP Dial-Up](#)